

FIG. 1

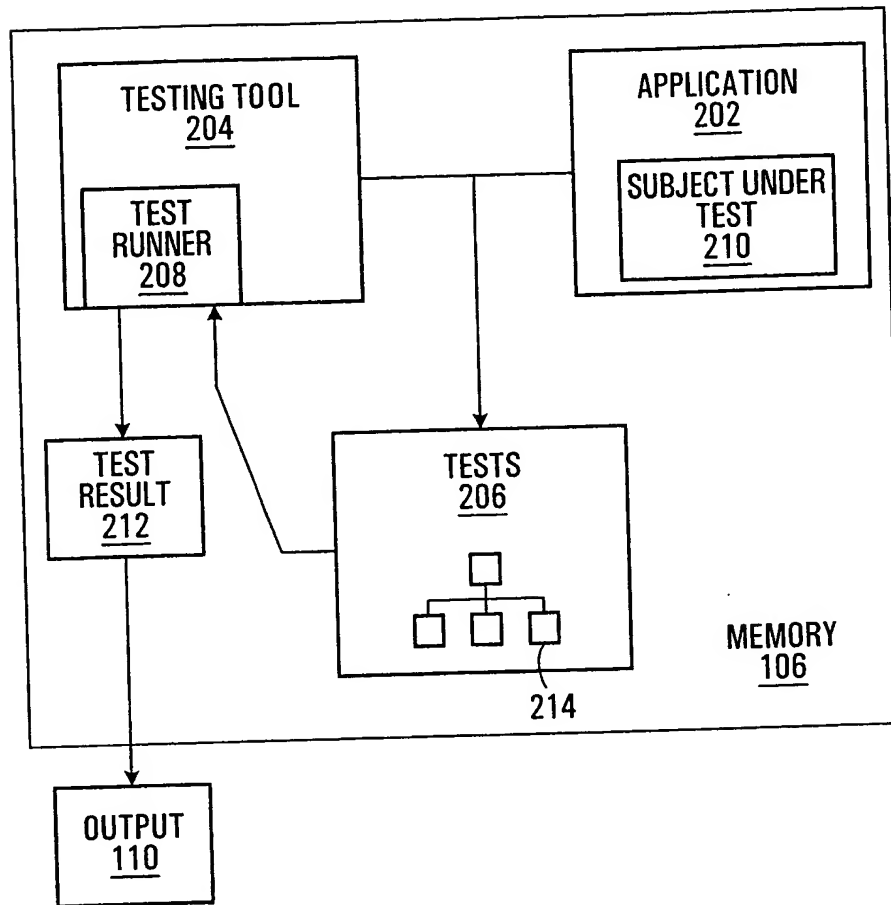


FIG. 2

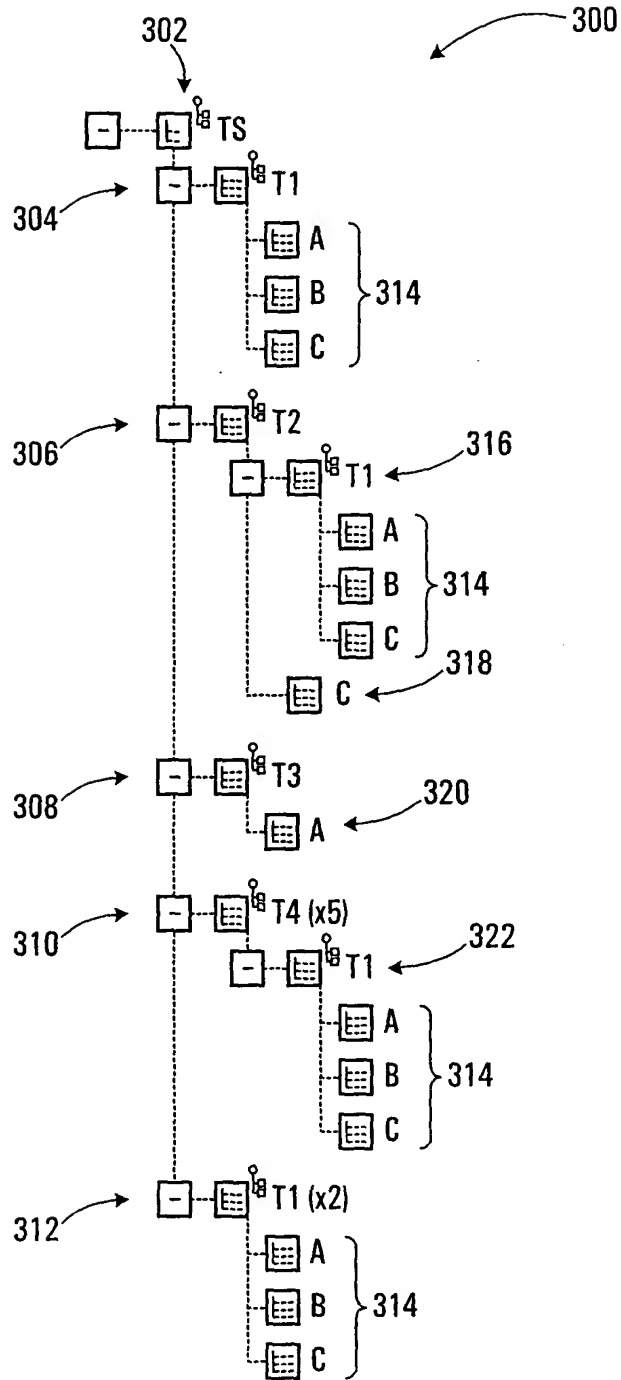


FIG. 3

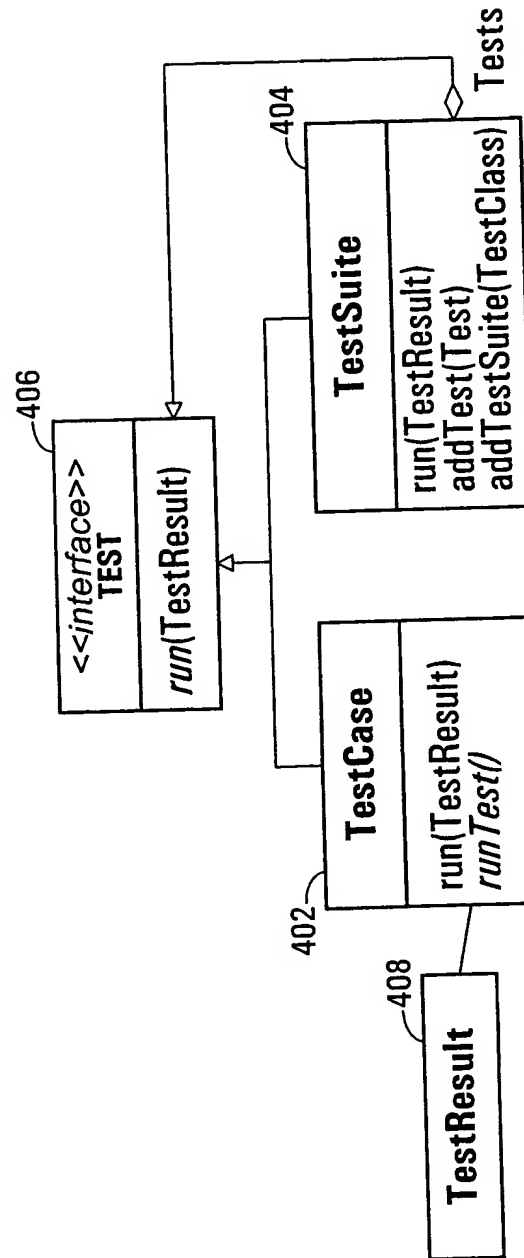


FIG. 4

FIG. 5

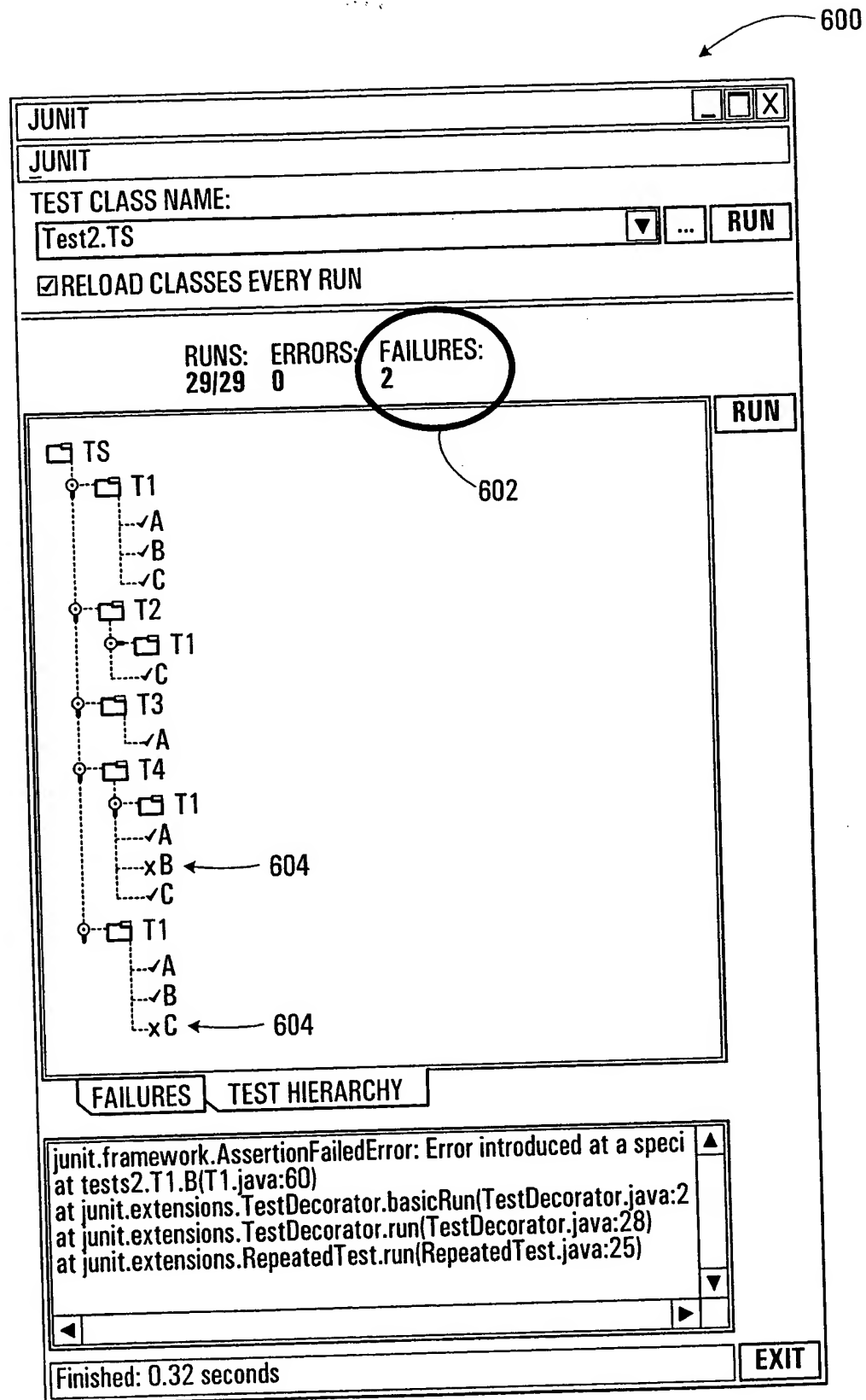


FIG. 6

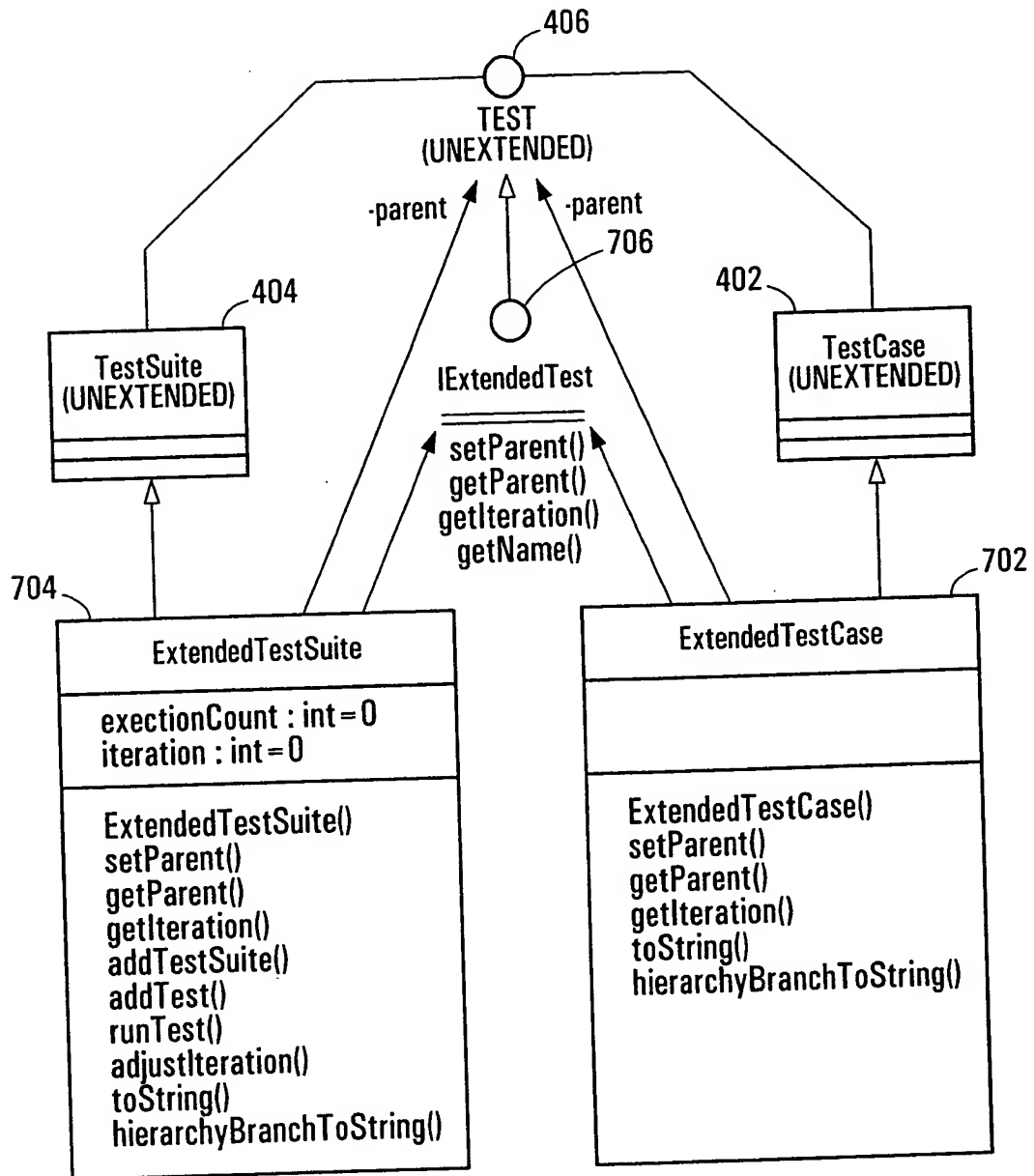
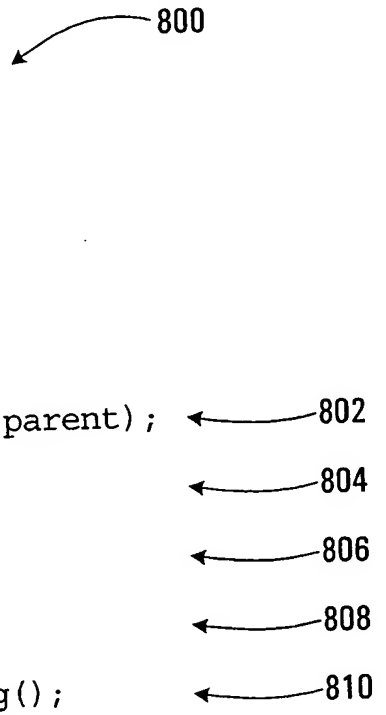


FIG. 7

+



```
1 package junit;
2
3 import junit.framework.Test;
4
5 interface IExtendedTest
6 extends Test
7 {
8     public void setParent(Test parent); ← 802
9
10    public Test getParent(); ← 804
11
12    public int getIteration(); ← 806
13
14    public String getName(); ← 808
15
16    public String toJUnitString(); ← 810
17 }
```

FIG. 8

+

+

```
1  package junit;
2
3  import junit.extensions.TestDecorator;
4  import junit.framework.Test;
5  import junit.framework.TestResult;
6  import junit.framework.TestSuite;
7
8  class ExtendedTestSuite
9  extends TestSuite implements IExtendedTest
10 {
11     private Test parent;
12
13     private int iteration = 0;
14     private int executionCount = 0;
15     private int previousParentIteration = 0;
16
17     public void setParent(Test parent) } 902
18     {
19         this.parent = parent;
20     }
21
22     public Test getParent() } 904
23     {
24         return parent;
25     }
26
27     public int getIteration() } 906
28     {
29         return iteration;
30     }
31
32     public void addTestSuite(Class testClass) } 908
33     {
34         addTest(new ExtendedTestSuite(testClass));
35     }
36
37     public void addTest(Test test)
38     {
39         super.addTest(test);
40
41         if (test instanceof TestDecorator)
42             test = ((TestDecorator) test).getTest();
43
44         if (test instanceof IExtendedTest)
45         {
46             IExtendedTest extendedTest =
47             (IExtendedTest) test;
48             extendedTest.setParent(this);
49         }
50
51     }
52
53     public void runTest(Test test, TestResult result) } 912
54     {
55         adjustIteration();
56         super.runTest(test, result);
57     }
```

FIG. 9A

+

+

```

58     protected void adjustIteration()
59     {
60         if((parent != null) && (parent instanceof
IExtendedTest))
61         {
62             int parentIteration =
((IExtendedTest)parent).getIteration();
63             if(parentIteration != previousParentIteration)
64             {
65                 executionCount = 0;
66                 previousParentIteration =
parentIteration;
67             }
68         }
69         executionCount++;
70         iteration = ((int) (executionCount /
this.testCount())) + 1;
71         if((executionCount % this.testCount()) == 0)
72             iteration--;
73     }
74 }
75
76     public String toJUnitString()
77     {
78         return super.toString();
79     }
80
81     public String toString()
82     {
83         return hierarchyBranchToString(this);
84     }
85
86     protected String hierarchyBranchToString(Test test)
87     {
88         if ((test != null) && (test instanceof
IExtendedTest))
89         {
90             IExtendedTest extendedProvider =
(IExtendedTest) test;
91             String hierarchy =
hierarchyBranchToString(extendedProvider.getParent());
92             if(hierarchy.length() > 0)
93                 hierarchy = hierarchy + ">";
94
95             hierarchy = hierarchy +
extendedProvider.getName();
96             if(extendedProvider.getIteration() > 0)
97                 hierarchy = hierarchy + "_" +
extendedProvider.getIteration();
98
99             return hierarchy;
100         }
101         if(test != null)
102             return "?>" + test.toString();
103         return "";
104     }
105 }
106 }

```

914

916

918

920

FIG. 9B

+

+

1000

```
1 package junit;
2
3 import junit.framework.Test;
4 import junit.framework.TestCase;
5
6 class ExtendedTestCase
7 extends TestCase implements IExtendedTest
8 {
9     private Test parent;
10
11     /**
12      * Constructor for JUnitTestCase.
13      * @param name
14      */
15     public ExtendedTestCase(String name)
16     {
17         super(name);
18     }
19
20     public void setParent(Test parent) } 1002
21     {
22         this.parent = parent;
23     }
24
25     public Test getParent() } 1004
26     {
27         return parent;
28     }
29
30     public int getIteration()
31     {
32         if((parent != null) && (parent instanceof
ExtendedTestSuite))
33         return ((ExtendedTestSuite)
parent).getIteration();
34
35         return -1;
36     }
37
38     public String toJUnitString()
39     {
40         return super.toString();
41     }
42
43     public String toString()
44     {
45         return hierarchyBranchToString(this);
46     }
47
48     protected String hierarchyBranchToString(Test test)
49     {
```

1006

FIG. 10A

+

```
50     if ((test != null) && (test instanceof IExtendedTest))
51     {
52         IExtendedTest extendedProvider = (IExtendedTest) test;
53         String hierarchy =
54             hierarchyBranchToString(extendedProvider.getParent());
55         if (hierarchy.length() > 0)
56             hierarchy = hierarchy + ">";
57
58         hierarchy = hierarchy + extendedProvider.getName();
59         if (extendedProvider.getIteration() > 0)
60             hierarchy = hierarchy + "_" + extendedProvider.getIteration();
61
62         return hierarchy;
63     }
64     if (test != null)
65         return "?>" + test.toString();
66
67     return "";
68 }
69 }
```

FIG. 10B

+

FIG. 11

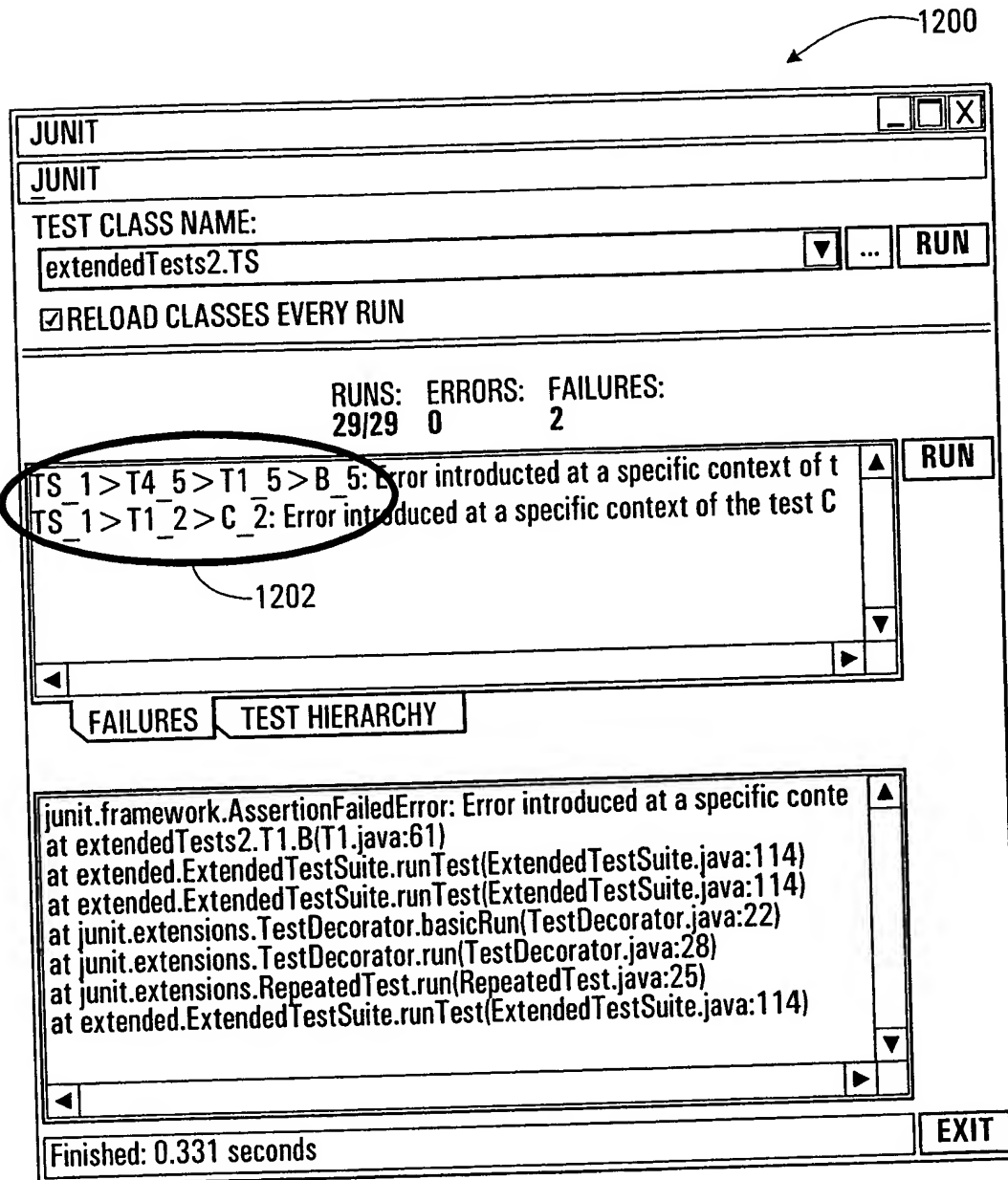


FIG. 12